



捷恩斯威  
JEANSWAY

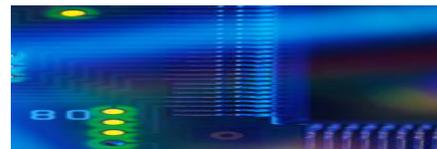


## LM4F 浮点运算应用

Let`s make your development easier!

**捷恩斯威科技，最专业的TI MCU方案设计商**

[www.jeansway.cn](http://www.jeansway.cn)



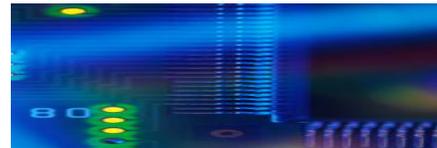
# 浮点数表示方法

## ❖ 定点格式

定点数指小数点在数中的位置是固定不变的，通常有定点整数和定点小数。在对小数点位置作出选择之后，运算中的所有数均应统一为定点整数或定点小数，在运算中不再考虑小数问题。

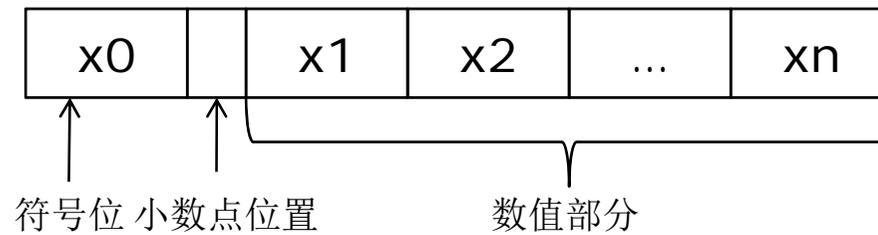
## ❖ 浮点格式

浮点数中小数点的位置是不固定的，用阶码和尾数来表示。通常尾数为纯小数，阶码为整数，尾数和阶码均为带符号数。尾数的符号表示数的正负；阶码的符号则表明小数点的实际位置

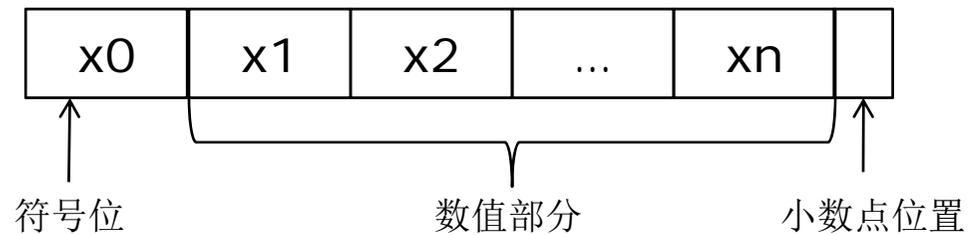


# 定点格式表示方法

## 定点小数



## 定点整数





# IEEE 754 浮点表示方法



N的实际值由下列式子表示：
$$n = (-1)^s \times m \times 2^e$$

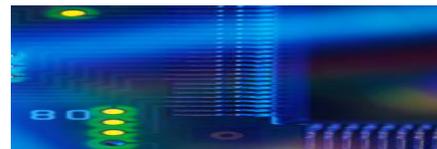
S: 表示符号位

E: 表示N的指数位

M: 表示N的尾数位

单精度浮点数: N共32位, 其中S占1位, E占8位, M占23位

双精度浮点数: N共64位, 其中S占1位, E占11位, M占52位



# 浮点运算的用途

1

电机控制

2

图像解码

3

音频解码

4

信号滤波

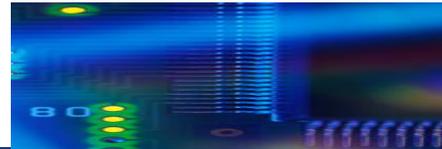
5

智能识别



## LM4F浮点运算简介

- ❖ 符合IEEE 754单精度浮点运算单元
- ❖ 32位指令集，具有单精度数据操作能力
- ❖ 硬件支持转换, 加, 减, 乘(乘累加), 除, 平方根
- ❖ 乘累加功能提高运算精度
- ❖ 32个专门的32位单精度寄存器，也可以按照16个又word寄存器方式寻址
- ❖ 可将FPU关闭以降低功耗。



捷恩斯威  
JEANSWAY

# Cortex-M指令集

|        |         |        |        |        |        |         |         |         |
|--------|---------|--------|--------|--------|--------|---------|---------|---------|
| PKH    | QADD    | QADD16 | QADD8  | QASX   | QADD   | QSUB    | QSAX    | QSUB    |
| QSUB16 | QSUB8   | SADD16 | SADD8  | SASX   | SEL    | SHADD16 | SHADD8  | SHASX   |
| SHSAX  | SHSUB16 | SHSUB8 | SMLABB | SMLABT | SMLATB | SMLATT  | SMLAD   | SMLALBB |
|        |         |        |        |        |        |         | SMLALBT | SMLALTB |
|        |         |        |        |        |        |         | SMLALTT | SMLALD  |
|        |         |        |        |        |        |         | SMLAWB  | SMLAWT  |
|        |         |        |        |        |        |         | SMLSD   | SMLSLD  |
|        |         |        |        |        |        |         | SMMLA   | SMMLS   |
|        |         |        |        |        |        |         | SMMUL   | SMUAD   |
|        |         |        |        |        |        |         | SMULBB  | SMULBT  |
|        |         |        |        |        |        |         | SMULTB  | SMULTT  |
|        |         |        |        |        |        |         | SMULWB  | SMULWT  |
|        |         |        |        |        |        |         | SMUSD   | SSAT16  |
|        |         |        |        |        |        |         | SSAX    | SSUB16  |
|        |         |        |        |        |        |         | SSUB8   | SXTAB   |
|        |         |        |        |        |        |         | SXTAB16 | SXTAH   |
|        |         |        |        |        |        |         | SXTB16  | UADD16  |
|        |         |        |        |        |        |         | UADD8   | UASX    |
|        |         |        |        |        |        |         | UHADD16 | UHADD8  |
|        |         |        |        |        |        |         | UHASX   | UHSAX   |
|        |         |        |        |        |        |         | UHSUB16 | UHSUB8  |
|        |         |        |        |        |        |         | UMAAL   | UQADD16 |
|        |         |        |        |        |        |         | UQADD8  | UQASX   |
|        |         |        |        |        |        |         | UQ8AX   | UQSUB16 |
|        |         |        |        |        |        |         | UQSUB8  | USAD8   |
|        |         |        |        |        |        |         | USADA8  | USAT16  |

|        |       |        |        |        |       |        |
|--------|-------|--------|--------|--------|-------|--------|
| ADC    | ADD   | ADR    | AND    | ASR    | B     | CLZ    |
| BFC    | BFI   | BIC    | CDP    | CLREX  | CBNZ  | CBZ    |
| CMN    |       |        |        | DBG    | EOR   | LDC    |
| LDMIA  |       |        |        | LDMDB  | LDR   | LDRB   |
| LDRBT  |       |        |        | LDRD   | LDREX | LDREXB |
| LDREXH |       |        |        | LDRH   | LDRHT | LDRSB  |
| LDRSBT |       |        |        | LDRSHT | LDRSH | LDRT   |
| MCR    |       |        |        | LSL    | LSR   | MLS    |
| MCRR   |       |        |        | MLA    | MOV   | MOVT   |
| MRC    |       |        |        | MRRC   | MUL   | MVN    |
| NOP    |       |        |        | ORN    | ORR   | PLD    |
| PLDW   |       |        |        | PLI    | POP   | PUSH   |
| RBIT   |       |        |        | REV    | REV16 | REVSH  |
| ROR    |       |        |        | RRX    | RSB   | SBC    |
| SBFX   |       |        |        | SDIV   | SEV   | SMLAL  |
| SMULL  |       |        |        | SSAT   | STC   | STMIA  |
| STMOB  |       |        |        | STR    | STRB  | STRBT  |
| STRD   | STREX | STREXB | STREXH | STRH   | STRHT | STRT   |
| SUB    | SXTB  | SXTH   | TBB    | TBH    | TEQ   | TST    |
| UBFX   | UDIV  | UMLAL  | UMULL  | USAT   | UXTB  | UXTH   |
| WFE    | WFI   | YIELD  | IT     |        |       |        |

|      |        |       |       |         |       |        |
|------|--------|-------|-------|---------|-------|--------|
| USAX | USUB16 | USUB8 | UXTAB | UXTAB16 | UXTAH | UXTB16 |
|------|--------|-------|-------|---------|-------|--------|

|       |      |       |       |      |       |      |       |       |
|-------|------|-------|-------|------|-------|------|-------|-------|
| VABS  | VADD | VCMP  | VCMPE | VCVT | VCVTR | VDIV | VLDM  | VLDR  |
| VMLA  | VMLS | VMOV  | VMRS  | VMSR | VMUL  | VNEG | VNMLA | VNMLS |
| VNMUL | VPOP | VPUSH | VSQRT | VSTM | VSTR  | VSUB |       |       |

Let`s make your development easier!



www.jeansway.cn



# 无FPU与有FPU加减运算比较

Float加法运算:  $fadd = a + b$ , ( $a = 6.7213, b = 3.4123$ )

|       | M4禁止FPU                          | M4使能FPU               |       |
|-------|----------------------------------|-----------------------|-------|
| 51条指令 | <code>__aeabi_fadd:</code>       | <code>VADD.F32</code> | 仅1条指令 |
|       | <code>PUSH {r4-r7}</code>        |                       |       |
|       | <code>EOR r2,r0,r1</code>        |                       |       |
|       | <code>LSRS r2,r2,#31</code>      |                       |       |
|       | <code>...</code>                 |                       |       |
|       | <code>ADD r1,r3,r1,LSL #1</code> |                       |       |
|       | <code>SUBS r0,r0,r2</code>       |                       |       |
|       | <code>B 0x000003FE</code>        |                       |       |
|       | <code>POP {r4-r7}</code>         |                       |       |
|       | <code>BX lr</code>               |                       |       |

Float减法运算:  $fsub = a - b$ , ( $a = 6.7213, b = 3.4123$ )

|       | M4禁止FPU                            | M4使能FPU               |       |
|-------|------------------------------------|-----------------------|-------|
| 68条指令 | <code>__aeabi_fsub:</code>         | <code>VSUB.F32</code> | 仅1条指令 |
|       | <code>EOR r2,r0,r1</code>          |                       |       |
|       | <code>AND r3,r2,#0x80000000</code> |                       |       |
|       | <code>PUSH {r4-r5}</code>          |                       |       |
|       | <code>...</code>                   |                       |       |
|       | <code>ADD r0,r4,r5,LSL #23</code>  |                       |       |
|       | <code>POP {r4-r5}</code>           |                       |       |
|       | <code>ADD r0,r0,r3</code>          |                       |       |
|       | <code>NOP.W</code>                 |                       |       |



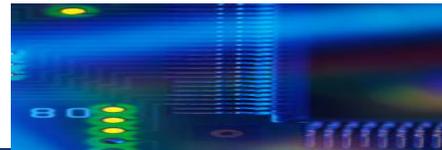
# 无FPU与有FPU乘除运算比较

Float乘法运算:  $fmul = a * b$ , ( $a= 6.7213, b= 3.4123$ )

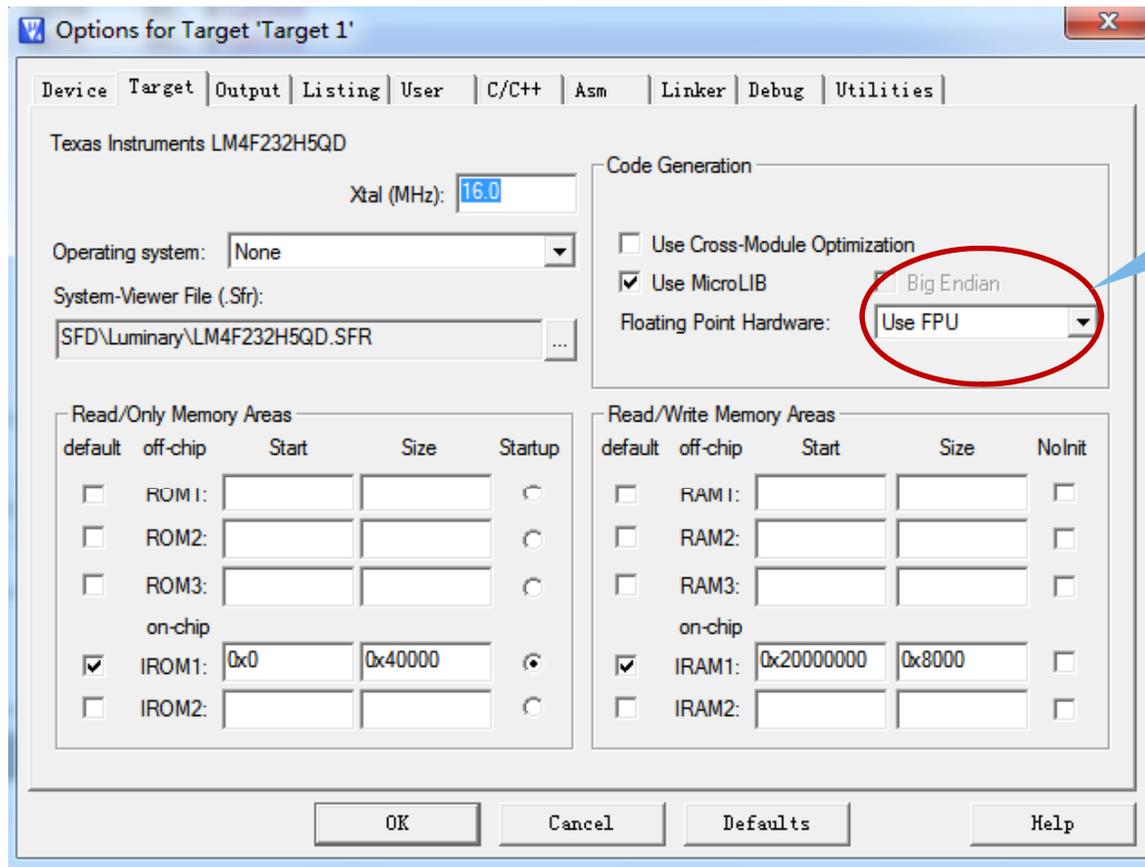
|       | M4禁止FPU               | M4使能FPU   |       |
|-------|-----------------------|-----------|-------|
| 49条指令 | __aeabi_fmul:         | VMUL. F32 | 仅1条指令 |
|       | EOR r2,r0,r1          |           |       |
|       | PUSH {r4,lr}          |           |       |
|       | AND r2,r2,#0x80000000 |           |       |
|       | ....                  |           |       |
|       | BGE 0x0000043C        |           |       |
|       | MOVS r0,#0x00         |           |       |
|       | ORRS r0,r0,r2         |           |       |
|       | POP {r4,pc}           |           |       |
|       |                       |           |       |

float除法运算:  $fdiv = a/b$ , ( $a= 6.7213, b= 3.4123$ )

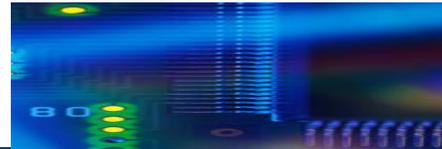
|        | M4禁止FPU               | M4使能FPU   |       |
|--------|-----------------------|-----------|-------|
| 140条指令 | __aeabi_fdiv:         | VDIV. F32 | 仅1条指令 |
|        | EOR r2,r0,r1          |           |       |
|        | AND r3,r2,#0x80000000 |           |       |
|        | PUSH {r4-r5}          |           |       |
|        | ...                   |           |       |
|        | ADD r0,r4,r5,LSL #23  |           |       |
|        | POP {r4-r5}           |           |       |
|        | ADD r0,r0,r3          |           |       |
|        | NOP.W                 |           |       |
|        |                       |           |       |



# LM4F的FPU体验



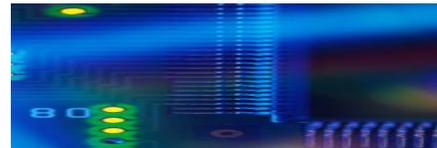
KEIL编译器使能FPU



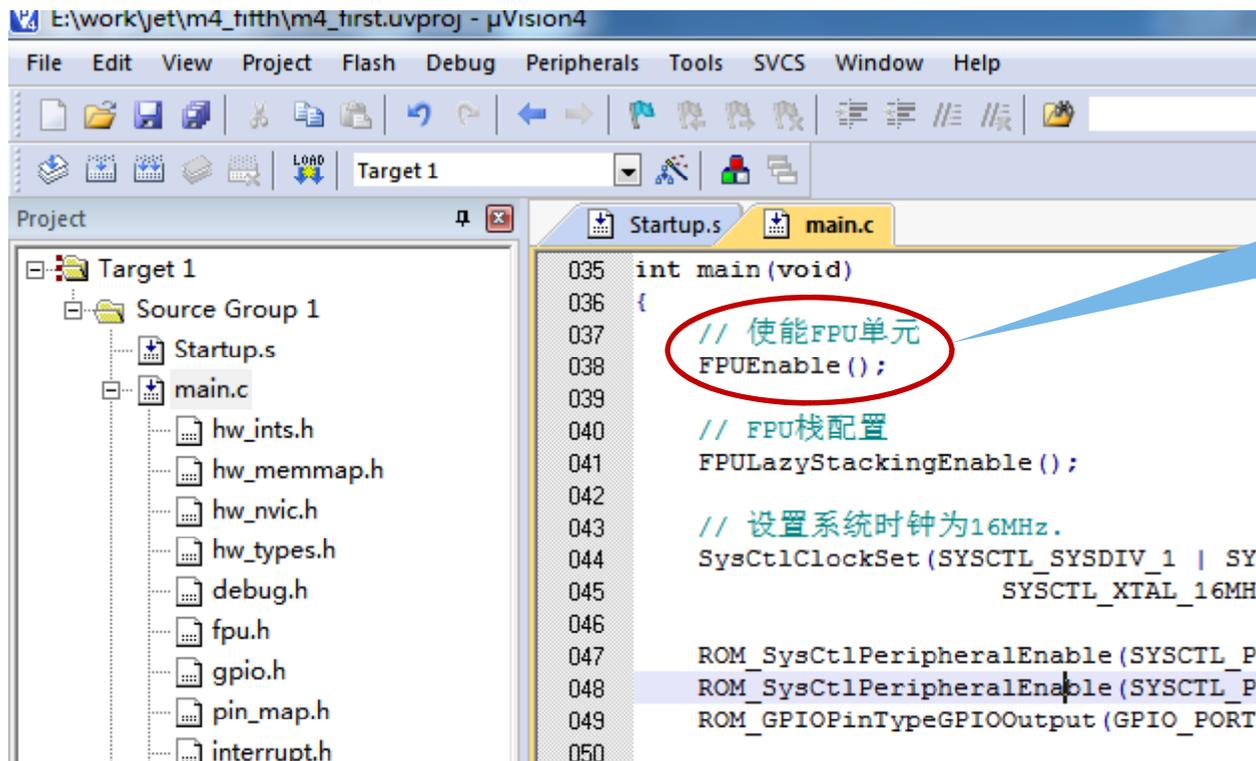
# LM4F的FPU使用

在KEIL编译器中自动生成Cortex-M4F FPU使能代码

```
Reset_Handler
;
; Enable the floating-point unit. This must be done here to handle the
; case where main() uses floating-point and the function prologue saves
; floating-point registers (which will fault if floating-point is not
; enabled). Any configuration of the floating-point unit using
; DriverLib APIs must be done here prior to the floating-point unit
; being enabled.
;
; Note that this does not use DriverLib since it might not be included
; in this project.
;
MOVW    R0, #0xED88
MOVW    R0, #0xE000
LDR     R1, [R0]
ORR     R1, #0x00F00000
STR     R1, [R0]
;
; Call the C library entry point that handles startup. This will copy
; the .data section initializers from flash to SRAM and zero fill the
; .bss section.
;
IMPORT  __main
B       __main
```

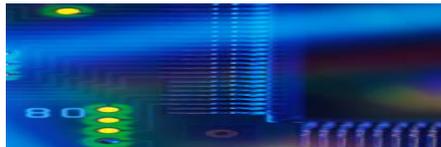


# 使用外设驱动库API 使能FPU



```
035 int main(void)
036 {
037     // 使能FPU单元
038     FPUEnable();
039
040     // FPU栈配置
041     FPULazyStackingEnable();
042
043     // 设置系统时钟为16MHz.
044     SysCtlClockSet(SYSCTL_SYSDIV_1 | SY
045                   SYSCTL_XTAL_16MH
046
047     ROM_SysCtlPeripheralEnable(SYSCTL_P
048     ROM_SysCtlPeripheralEnable(SYSCTL_P
049     ROM_GPIOPinTypeGPIOOutput(GPIO_PORT
050
```

使能Cortex-M4F FPU



# 使用FPU进行低通滤波处理

Assembly code snippet:

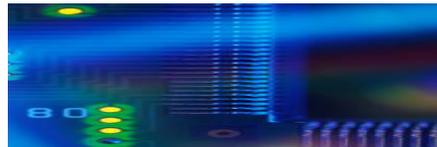
```
99:   for (i=0; i<=2; i++)
100:   {
0x000004A8 F04F0400 MOV     r4, #0x00
0x000004AC E029      B       0x00000502
101:   a[i]=a[i]/t;
0x000004AE EB0500C4 ADD     r0, r5, r4, LSL #3
0x000004B2 ED900B00 VLDR   d0, [r0, #0x00]
0x000004B6 EEB01A40 VMOV.F32 s2, s0
0x000004BA EEF01A60 VMOV.F32 s3, s1
0x000004D2 EEF0CA60 VMOV.F32 s25, s1
0x000002D6 ED9F0BD2 VLDR   d0, [pc, #0x348]
0x000002DA EC532B10 VMOV   r2, r3, d0
0x000002DE EC510B1C VMOV   r0, r1, d12
0x000002E2 F000FA67 BL.W   __aeabi_dmul (0x000007B4)
0x000002E6 EC410B10 VMOV   d0, r0, r1
0x000002EA ED8D0B06 VSTR   d0, [sp, #0x18]
84:   omp=tan(wp/2.0);
0x000002EE ED9F0BCA VLDR   d0, [pc, #0x328]
0x000002F2 EC532B10 VMOV   r2, r3, d0
0x000002F6 ED9D0B06 VLDR   d0, [sp, #0x18]
0x000002FA EC510B10 VMOV   r0, r1, d0
0x000002FE F000FA67 BL.W   __aeabi_ddiv (0x000008B4)
```

C code snippet:

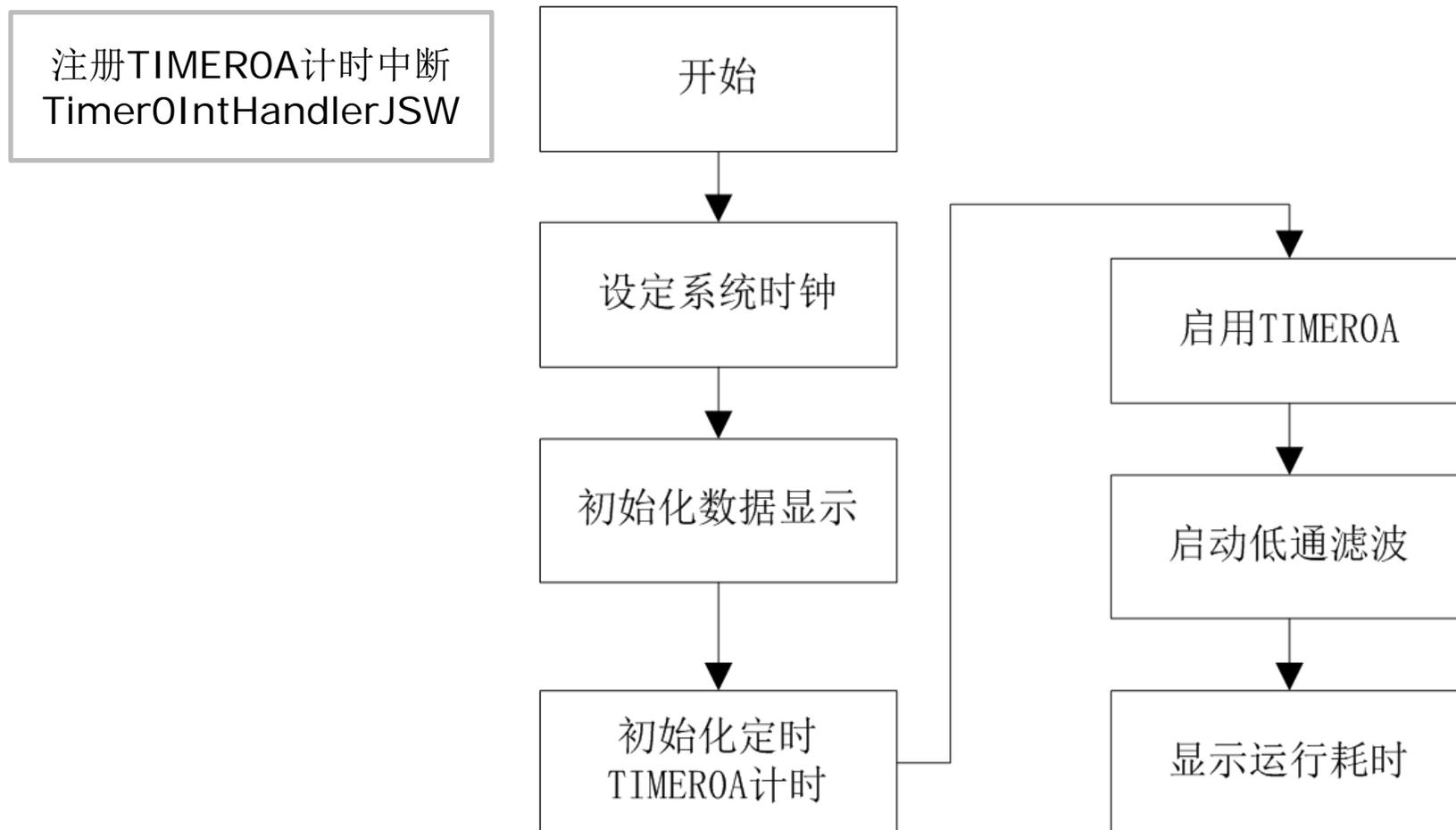
```
075 //
076 // 低通滤波计算函数
077 //
078 //*****
079 void biir2lpdes(double nlpass, double nlstop, double a[], double b[])
080 {
081     int i,u,v;
082     double wp,omp,gsa,t;
083     wp=nlpass*2*pi;
084     omp=tan(wp/2.0);
085     gsa=omp*omp;
```

浮点指令

C库函数



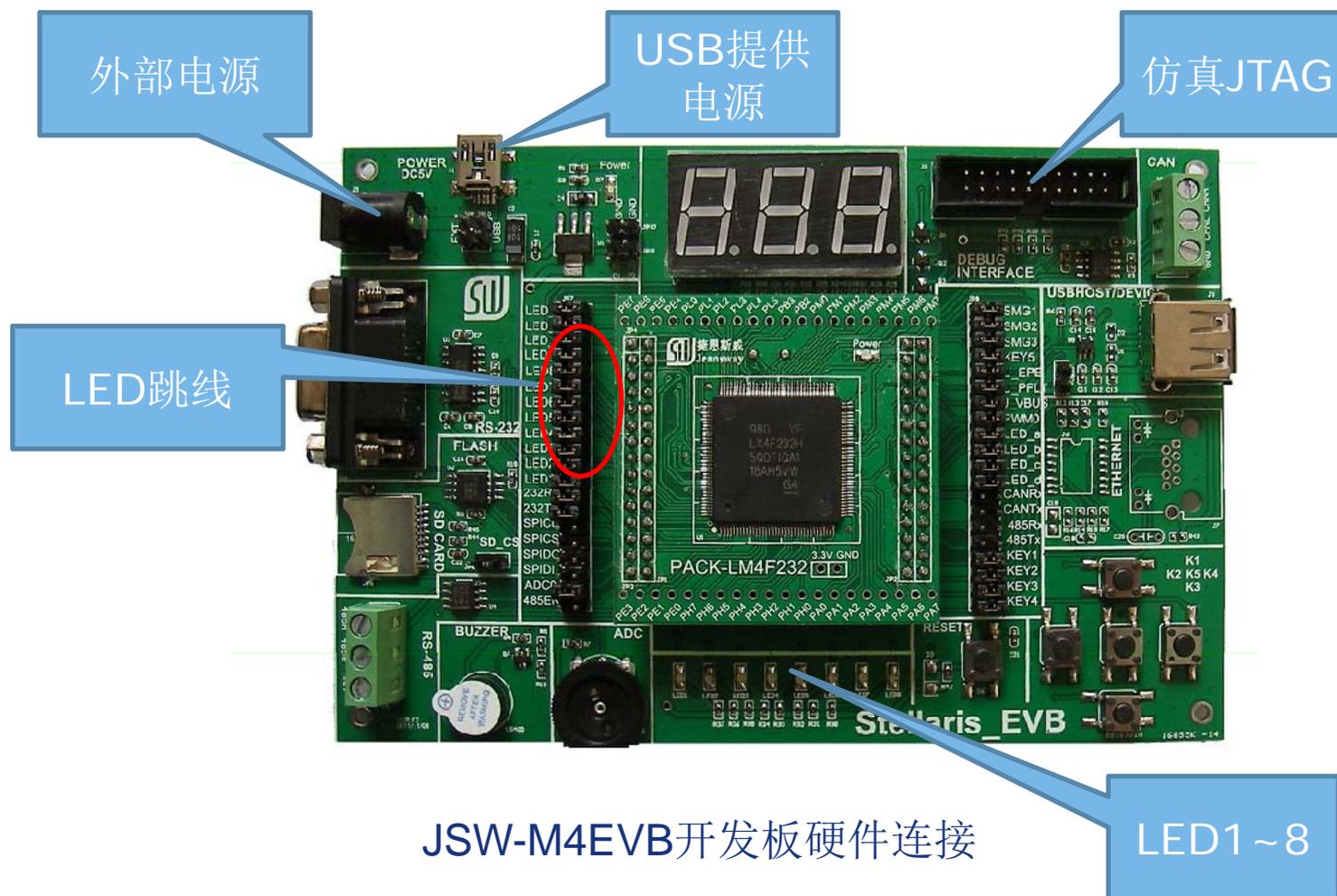
# LM4F的FPU体验流程图





捷恩斯威  
JEANSWAY

# 硬件连接



Let`s make your development easier!



捷恩斯威  
JEANSWAY

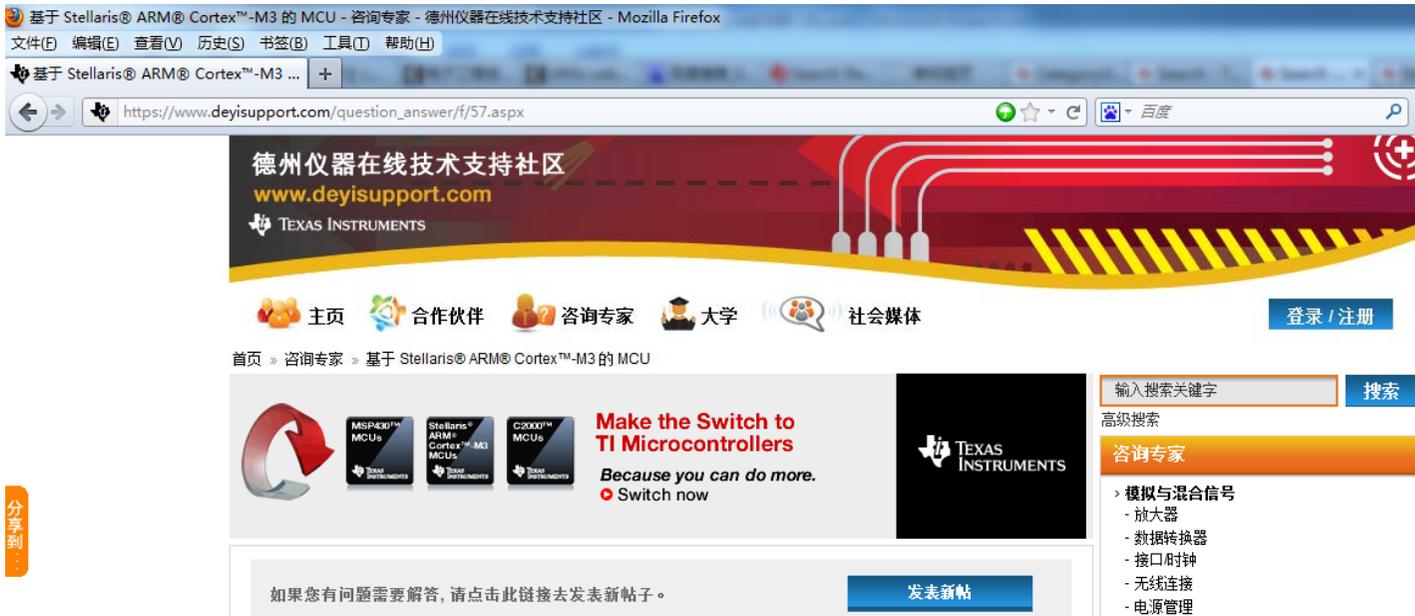
www.jeansway.cn



# 在线技术支持

<https://www.deyisupport.com/>

[https://www.deyisupport.com/question\\_answer/f/57.aspx](https://www.deyisupport.com/question_answer/f/57.aspx)



德州仪器在线技术支持社区  
www.deyisupport.com  
TEXAS INSTRUMENTS

主页 合作伙伴 咨询专家 大学 社交媒体 登录 / 注册

首页 » 咨询专家 » 基于 Stellaris® ARM® Cortex™-M3 的 MCU

Make the Switch to TI Microcontrollers  
Because you can do more.  
Switch now

输入搜索关键字 搜索

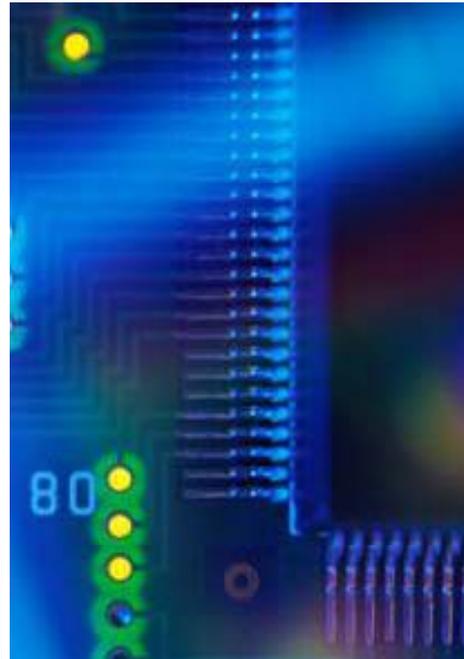
高级搜索

咨询专家

- 模拟与混合信号
- 放大器
- 数据转换器
- 接口时钟
- 无线连接
- 电源管理

如果您有问题需要解答, 请点击此链接去发表新帖子。 发表新帖

分享到...



**Thank You !**

[www.jeansway.cn](http://www.jeansway.cn)